Arbeitsblatt Nr.11: Entfernung messen

Aufgabe: Mit den Ultraschallsensor HC-SR04 und einem Arduino Mikrocontrollerboard soll eine Entfernung gemessen und mit dem „Serial-monitor“ angezeigt werden.

Material:

- Arduino-Board
- HC-SR04 Ultraschallsensor
- Breadboard
- Steckkabel

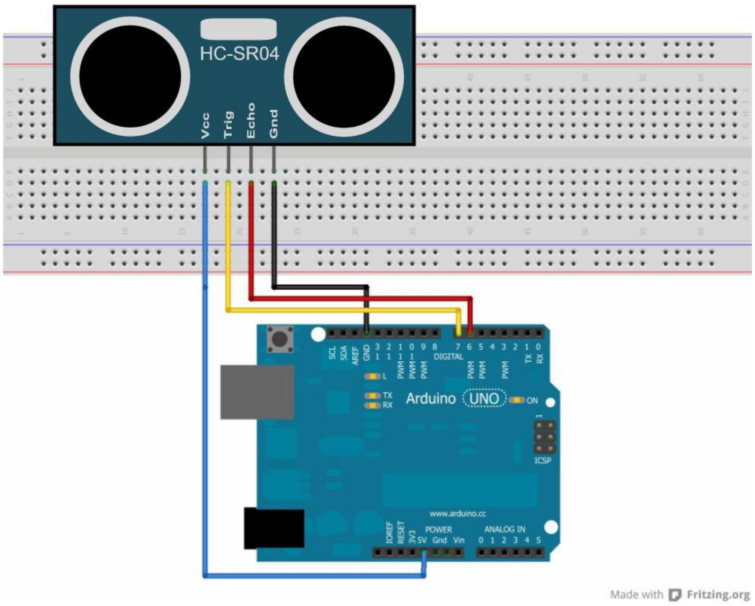
In dieser Anleitung lernst du:

- Ultraschallsensor anschließen
- PulseIn
- Messwerte mathematisch nutzbar machen
- Schallgeschwindigkeitskonstante nutzen

Wie funktioniert der Ultraschallsensor?
Der Sensor hat vier Kontakte 1) 5V(+) 2) GND (–) 3) Echo 4) Trigger

Die Anschlüsse 5V und GND verstehen sich von selbst, sie versorgen den Sensor mit Energie.
Der Pin „trigger“ bekommt vom Mikrocontroller-Board ein kurzes Signal (5V), wodurch eine Schallwelle vom Ultraschallsensor ausgelöst wird. Sobald die Schallwelle gegen eine Wand oder sonstigen Gegenstand stößt, wird sie reflektiert und kommt wieder zum Ultraschallsensor zurück. Sobald der Sensor diese zurückgekehrte Schallwelle erkennt, sendet der Sensor auf dem „echo“ Pin ein 5V Signal an das Mikrocontroller-Board. Dieser misst dann lediglich die Zeit zwischen dem Aussenden und der Rückkehr der Schallwelle und rechnet diese Zeit dann in eine Entfernung um. Der verwendete Befehl für die Zeitmessung lautet „PulseIn“, wobei der vorletzte Buchstabe ein großes „ i „ ist. (Häufige Fehlerquelle beim abtippen des Codes)

Aufbau:



Sketch:

```
int trigger=7;                //Trigger-Pin des Ultraschallsensors an Pin7 des
                              Arduino-Boards
int echo=6;                   //Echo-Pim des Ultraschallsensors an Pin6 des
                              Arduino-Boards
long dauer=0;                 //Das Wort dauer ist jetzt eine Variable, unter
                              der die Zeit gespeichert wird, die eine
                              Schallwelle bis zur Reflektion und zurück
                              benötigt. Startwert ist hier 0.
long entfernung=0;            //Das Wort „entfernung“ ist jetzt die variable,
                              unter der die berechnete Entfernung gespeichert
                              wird. Info: Anstelle von „int“ steht hier vor den
                              beiden Variablen „long“. Das hat den Vorteil, dass
                              eine größere Zahl gespeichert werden kann.
                              Nachteil: Die Variable benötigt mehr Platz im
                              Speicher des Mikrocontrollers.
void setup()                  //Beginn des Setups
{
Serial.begin (9600);          //Serielle kommunikation starten, damit man sich
                              später die Werte am serial monitor ansehen kann.
pinMode(trigger, OUTPUT);     //Trigger-Pin ist ein Ausgang
pinMode(echo, INPUT);         //Echo-Pin ist ein Eingang
}
void loop()
{
digitalWrite(trigger, LOW);   //Hier nimmt man die Spannung für kurze Zeit vom
                              Trigger-Pin, damit man später beim senden des
                              Trigger-Signals ein rauschfreies Signal hat.
Delay(5);                     //Dauer: 5 Millisekunden
digitalWrite(trigger, HIGH);  //Jetzt sendet man eine Ultraschallwelle los.
Delay(10);                    //Dieser „Ton“ erklingt für 10 Millisekunden.
digitalWrite(trigger, LOW);   //Dann wird der „Ton“ abgeschaltet.
dauer = pulseIn(echo, HIGH);  //Mit dem Befehl „pulseIn“ zählt der
                              Mikrokontroller die Zeit in Mikrosekunden, bis der
                              Schall zum Ultraschallsensor zurückkehrt.
entfernung = (dauer/2) * 0.03432;
//Nun berechnet man die Entfernung in Zentimetern. Man teilt zunächst die Zeit
durch zwei (Weil man ja nur eine Strecke berechnen möchte und nicht die Strecke
hin- und zurück). Den Wert multipliziert man mit der Schallgeschwindigkeit in
der Einheit Zentimeter/Mikrosekunde und erhält dann den Wert in Zentimetern.

if (entfernung >= 500 || entfernung <= 0) //Wenn die gemessene Entfernung über
                                          500cm oder unter 0cm liegt,…
{
Serial.println("Kein Messwert"); //dann soll der serial monitor ausgeben „Kein
                                 Messwert“, weil Messwerte in diesen Bereichen
                                 falsch oder ungenau sind.
}
else                          //Ansonsten…
{
Serial.print(entfernung);     //…soll der Wert der Entfernung an den serial
                              monitor hier ausgegeben werden.
Serial.println(" cm");        //Hinter dem Wert der Entfernung soll auch am
                              Serial Monitor die Einheit "cm" angegeben werden.
}
delay(1000);                  //Das delay von einer Sekunde sorgt in ca. jeder
                              neuen Sekunde für einen neuen Messwert.
}
```

Erweiterung des Programms:

Wenn ein Abstand unter 80cm gemessen wird, soll ein Piezo-Lautsprecher piepsen.

```
int trigger=12;
int echo=13;
long dauer=0;
long entfernung=0;
int piezo=5;                  //Das Wort piezo ist jetzt die Zahl 5

void setup()
{
Serial.begin (9600);
pinMode(trigger, OUTPUT);
pinMode(echo, INPUT);
pinMode(piezo, OUTPUT);       //Der Pin5 soll ein Ausgang sein, da der
                              Lautsprecher eine Spannung benötigt um einen Ton
                              zu erzeugen.
}
void loop()
{
digitalWrite(trigger, LOW);
delay(5);
digitalWrite(trigger, HIGH);
delay(10);
digitalWrite(trigger, LOW);
dauer = pulseIn(echo, HIGH);
entfernung = (dauer/2) * 0.03432;
if (entfernung >= 500 || entfernung <= 0)
{
Serial.println("Kein Messwert");
}
else
{
Serial.print(entfernung);
Serial.println(" cm");
}
//Es wird eine weitere IF-Bedingung erstellt:
if (entfernung <= 80)         //Wenn der Wert für die Entfernung unter oder
                              gleich 80 ist, dann...
{
digitalWrite(piezo,HIGH);     //...fange an zu piepsen.
}
else                          //Und wenn das nicht so ist...
{
digitalWrite(piezo,LOW);      //...dann sein leise.
}
delay(1000);
}
```

Erweiterung: Rückfahrwarner

Aufgabe: Nutze die erlernten Dinge, um einen Rückfahrwarner zu konstruieren. Ein Beispiel dazu ist auf der Internetseite www.funduino.de verfügbar.